Dark Mode Implementation

Introduction

In the Play+ ecosystem, a smooth, comfortable experience in all environments is a core design goal. Whether your users prefer a darker UI or are working late into the evening, Play+ adapts effortlessly. Our Dark Theme isn't bolted on—it's built in, and designed to maintain your brand's distinctiveness and readability.

You don't need to define a separate dark theme. Just define your light theme as usual, and Play+ takes care of the rest.

Why There's No Separate Dark Theme

Traditionally, dark mode meant duplicating styles, increasing complexity and potential bugs. In Play+, that duplication is unnecessary.

With a single _dark.css file, Play+ intelligently derives a dark variant of your theme. It respects your brand's tone and automatically adjusts colors, backgrounds, and contrast to suit dark contexts—without disrupting your design language.

How It Works

- Define Your Theme Provide your core tokens in _default.css , including brand colors, text styles, and backgrounds.
- Dark Theme Engine Kicks In Play+ processes this theme and generates a full dark mode version via _dark.css . No extra configuration required.
- System Preference Detection When a user's system is set to dark mode, Play+ switches automatically using the prefers-color-scheme: dark media query.

What Gets Transformed?

Surfaces

Light backgrounds are softened to rich dark grays (e.g., --global-color-gray-900), avoiding pure black. Secondary layers maintain visual depth.

Text

Text colors are lightened to remain readable on dark surfaces, and accessibility contrast is recalculated.

Brand Colors

Bright brand colors are adapted—desaturated or brightened if needed—to reduce harsh contrast. Related tokens like --color-text-on-brand-primary adjust accordingly.

Disabling Automatic Detection

To ignore system preferences and apply themes manually, update your config:

This disables automatic switching. You can then manage the theme explicitly via toggle or app logic.

Overriding the Defaults

Most themes work great with automatic derivation. But if you need to override a specific value, just add a custom token in your theme file:

This gives you precise control when needed—without losing the benefits of derivation.

Manual Theme Toggle

You can give users a manual theme toggle in your UI using the data-theme attribute on the <html> element. This is especially useful if you've disabled automatic OS detection.

For Angular

Add this logic to a shared service or component, such as theme-toggle.component.ts :

Template:

File Summary

- Angular : Implement in a dedicated theme-toggle.component.ts with corresponding HTML

You can give users a theme toggle in your UI using the data-theme attribute on <html> :

This empowers users and complements OS-level preference detection.

Developer Checklist

- Define a complete light theme in _default.css
- Let Play+ derive the dark variant automatically via _dark.css
- Preview before overriding
- Use custom overrides sparingly
- Confirm contrast accessibility if overridden
- Offer a user toggle if needed

Conclusion

With Play+, dark mode is automatic, accessible, and brand-aware. There's no need to manage two sets of styles or worry about visual quality. One well-defined theme is all it takes to deliver a polished experience—day or night.